

Chapter 1: Introduction

The IVR 2.0/I SQL Server option provides you with additional tools to develop effective voice messaging and interactive voice response (IVR) application solutions. The SQL Server interfaces with several SQL-based ANSI-compliant relational database management systems (DBMS) that allow your applications to retrieve, insert, update, and delete information stored in the supported databases.

This chapter contains the following:

- An overview of the SQL server
- Getting started
- Planning your application

An overview of the SQL server

IVR 2.0/I already supports application access to an information database and provides a method for building a simple database with the System Database Editor (SDE). You may, however, want to develop applications that can access SQL-based relational databases. The SQL Server allows IVR 2.0/I applications to access and update SQL databases. The SQL Server provides this functionality by adding the following ready-to-use cells that you can include in your application:

- QCNT — SQL Select Count Cell
- QDEL — SQL Delete Cell
- QINS — SQL Insert Cell
- QSEL — SQL Select Cell
- QUPD — SQL Update Cell

These cells execute standard SQL COUNT, SELECT, INSERT, UPDATE, and DELETE statements. Each cell corresponds to one SQL transaction. You can define access to database tables or views from within your applications. To make more complex queries, you can use traditional embedded SQL for C (ESQL/C) with IVR 2.0/I User cells.

The SQL Server is designed for an open environment and can interface with Ingres, Oracle, Sybase, and Informix. The following are currently supported:

- Ingres 6.4 (or newer) on SCO 3.0
- Oracle 7.0 (or newer) or 7.1 (or newer) on SCO 3.0
- Informix 5.0 (or newer) on SCO 3.0
- Sybase 10.0 (or newer) on SCO 3.0

Differences between SQL and the SQL server cells

To manipulate information in a database, you normally type one or more SQL statements at your keyboard, or include embedded SQL commands in a C program. The IVR 2.0/I SQL server cells are designed to execute SQL statements as well, but from within a IVR 2.0/I application.

When you create an SQL cell, you are specifying the same information as contained in a normal SQL statement, as shown in Figure 1-1. The cells are simpler to use since there is no need to worry about grammatically correct SQL syntax. All you need to reference is the correct table and column names. The SQL server executes the correct ANSI SQL commands based on the cell parameters that you specify.

The differences between SQL statements and the SQL Server cells follow:

- Only one table per cell is allowed. Consequently, to perform a join, you should build a view, and then reference the name of the view in the appropriate cell's table name parameter. Finally, build a view using your RDBMs.
- Complex queries, such as subqueries, are not supported (except as noted in the previous bullet item, when building a view).
- SQL expression syntax is not completely supported.
- Each cell represents one SQL transaction; you cannot roll back the cell transaction once the cell has completed processing. Each SQL statement is committed if it is successful, and rolled back if it fails.
- The QSEL (SQL select) cell currently returns only the first row of a matched data set.

Figure 1-1
Coding an SQL Statement versus an SQL Server Cell

SELECT Statement
 SQL Statement { SELECT A, B
 FROM TABLE 1
 WHERE X="1000"

SQL Server Cell

↕

QSEL Parameters

Cell #1 **QSEL SQL Select**

SQL Statement: Select a, b

Comments:

Call Audit Enabled? ☒ Yes ☐ No

Call Audit Information: DIGITS

SQL Table Name: TABLE_1

Column and Buffer Table

Column	Buffer
A	DATA EXCHANGE #1
B	DATA EXCHANGE #2
	DATA EXCHANGE #1
	DATA EXCHANGE #1

More Less

Where Clause

Logical	's	Column	Type	Operator	Value	's
	X		String	=	"1000"	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	

More Less

Apply Cancel Help

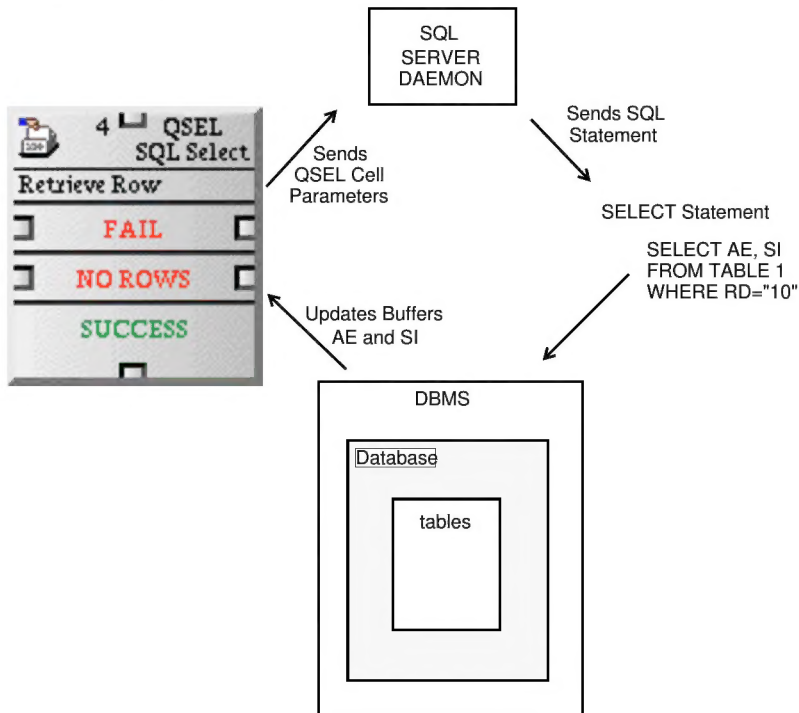
Running in the background

When you boot your system, IVR 2.0/I automatically loads the SQL server as a daemon process (or as middleware if you are using a remote SQL server). The server manages all requests by callers for information stored in the database and executes the corresponding SQL statements in the database management system (DBMS) as illustrated in Figure 1-2. The data that the caller can manipulate is determined by how you define the SQL cells in your application and the privileges you grant to the database user.

For Informix and Sybase, the applications run in the default tablespace defined for the user “vad” with a password of “vad1”. For Ingres, the applications run as the operating system user specified in the default cell. For Oracle, the applications run in the default tablespace defined for the user with a password specified in the default cell.

For both Oracle and Ingres, the password specified in the default cell must be correct. Consult the IVR 2.0/I transaction log verify the accuracy of your password.

Figure 1-2
Managing database queries from callers



In this illustration, when the QSEL (SQL SELECT) cell is processed during execution of the application, the cell parameters are passed to the SQL server daemon. The daemon process submits the corresponding SQL SELECT statement to the DBMS and retrieves the first row that matches the statement's selection criteria. The daemon process updates the output buffers identified by the QSEL cell, and the application continues processing with the next cell.

Getting started

Before you run an application that accesses an SQL-based database, you must meet the following prerequisites:

- Install the Database Management System (DBMS) on the application processor.
- Load (or create) the database that your application will access (or create).
- Create the necessary objects (for example, tables, views).
- Define or create the authorized user account(s), if necessary.
- Grant user access privileges to db objects.

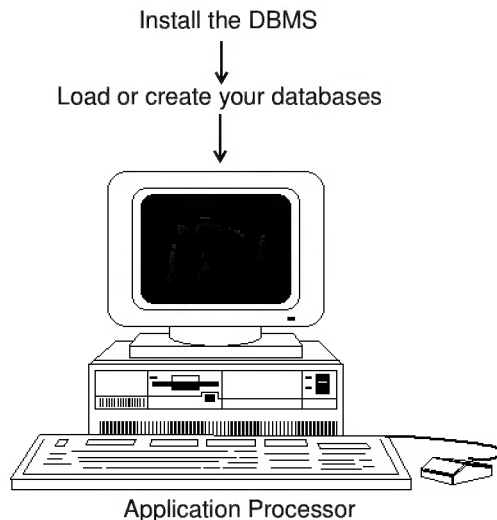
Installing the database management system

You can develop an application that manipulates data in a database, but the application will be useless without the database. IVR 2.0/I SQL Server is compatible with the following database management systems, or the specified platforms, that support ANSI SQL:

- Informix
- Ingres
- Oracle
- Sybase

You must purchase your DBMS separately from IVR 2.0/I and install it on the application processor.

Figure 1-3
Install your DBMS and load the data



Creating or updating the database

Once the DBMS has been installed on the application processor, use the tools associated with your DBMS to create or update the appropriate tables and views. (Refer to the Database Administrator guide accompanying your DBMS, or consult your company's Database Administrator.)

Setting up the user account

The only users who can access your database are those with authorized user accounts. For Informix and Sybase, the applications run in the default tablespace defined for the user "vad" with a password of "vad1". For Ingres, the applications run as the operating system user specified in the default cell. For Oracle, the applications run in the default tablespace defined for the user with a password specified in the default cell.

For both Oracle and Ingres, the password specified in the default cell must be correct. Consult the IVR 2.0/I transaction log verify the accuracy of your password.

Granting access privileges

To ensure that callers can access specific data while preserving the security of the database, the Database Administrator should grant the appropriate access privileges to those tables and views in the database that may be accessed by the user and the user account running IVR 2.0/I.

Planning your application

Before you start building your application, you should consider the following:

- What information do you want callers to be able to manipulate?
- Where is this information stored (that is, which database and tables)?
- Should callers be able to retrieve, insert, update, and/or delete data?

Once you have the names of the databases, tables, and columns where the data resides and have granted the appropriate privileges to the IVR 2.0/I accounts, you are ready to build your application.

Figure 1-4 shows an application that inserts, deletes, or updates information into a database based on a comparison made of the caller's input against the contents of several buffers.

Figure 1-4
Sample application showing Select, Update, Delete, and Insert

